Mistakes make us better–Car price prediction

1st Hu Silan, 2nd Pang Weicong, 3rd Yang Jiajun, 4th Huang Yulin
School of Computing, NUS

***Abstract*—Accurately predicting resale prices in the used car market can significantly aid both sellers and buyers in making informed decisions. This paper presents a comprehensive machine learning framework for forecasting used car prices in Singapore, utilizing data from SGCarMart with 29 attributes across 25,000 training records and 10,000 test records. We applied and compared four state-of-the-art gradient boosting models (XGBoost, LightGBM, CatBoost, and Gradient Boosting) and conducted an in-depth error analysis. Through error clustering and feature correlation analysis, we identified key performance patterns and feature impacts across error ranges. To improve prediction accuracy, we developed an ensemble strategy combining specialized models per error category, resulting in a 7% RMSE reduction compared to individual models. Furthermore, by integrating residual errors into model retraining, our final approach demonstrated a further 5% improvement. This study provides a scalable and generalizable framework for price prediction with significant implications for various regression tasks.**

I. Motivation and Introduction

The accurate prediction of used car resale prices contributes to market efficiency, benefiting sellers who want to achieve optimal prices and buyers who need fair valuations. For industry stakeholders, precise pricing models enhance inventory management and business strategy. In conclusion, the analysis provides insights into value determinants in the used car market and thus serves both academic understanding and practical application.

Our study develops a regression model using SGCarMart's historical transaction data, comprising 29 attributes across 25,000 training records and 10,000 test records. The goal of this task is to predict the resale price of a car based on its properties (e.g., make, model, mileage, age, power, etc). with model performance measured using RMSE.

II. Exploratory data analysis

A. Distribution of Car Resale Prices

There are only 3427 different price values. We take a look at the distribution of prices and find a strongly left-skewed distribution (Fig. 1). This indicates that some cars are way more expensive than others, so we can stratify by price. We note that log price is approximately normally distributed. This is a challenge because the metric is RMSE and it strongly penalizes outliers.

B. Numerical Data

The numerical values include listing_id, manufactured, curb_weight, power, engine_cap, no_of_owners, depreciation, coe, road_tax, dereg_value, mileage, omv and arf. We first performed a statistical analysis to explore our numerical data (Table 1).

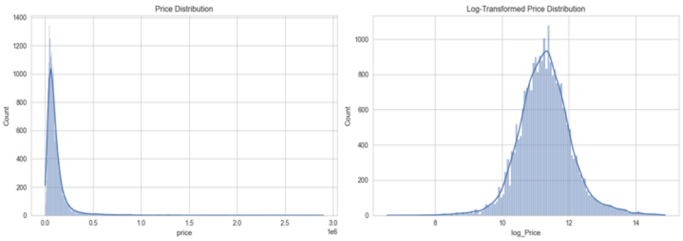

Fig. 1. The distribution of price and log-transformation price

TABLE I
Statistics of numerical data related to car prices.

Column	Count	Mean	Std	Min	Max	Null
manufactured	71	2016	5.72	1929	2024	7
curb_weight	973	1581	838	435	25620	307
power	311	141	82	0	735	2640
engine_cap	380	2059	1073	0	15681	596
no_of_owner	6	2	1	1	6	18
depreciation	3937	20466	19769	1770	607730	507
coe	2605	48935	21584	2605	158804	0
road_tax	609	1462	1377	50	12375	2632
dereg_value	18742	46037	61355	11	1114652	220
mileage	6449	85930	51625	1	129000	5304
omv	14398	41464	48375	426	811764	64
arf	14397	44727	79113	23	1491920	174

We plot a scatter plot between numerical data and prices (Fig. 2) to qualitatively explore the correlation. And we find attributes like omv, arf, dereg_value, power, engine_cap, and coe have strong positive correlations with price, while mileage has a clear negative correlation. Other attributes show weaker or negligible relationships. These insights can guide feature selection in predicting car prices.

To quantify the relationship, we developed a heatmap (Fig. 3) to investigate the relationship of each attribute with price and also the interrelation among attributes. If many features are highly correlated with each other(e.g., correlation coefficients close to 1 or -1), it may lead to issues of multicollinearity. Accordingly, we may consider dropping, merging highly correlated features or use the relationship to fill missing values in further analysis.

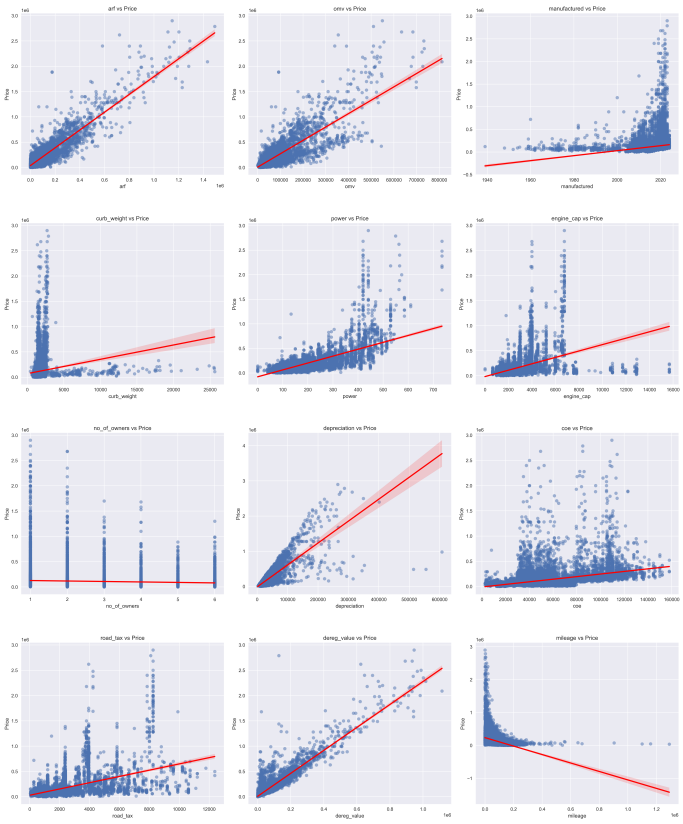

Fig. 2. A scatter plot between numerical data and prices

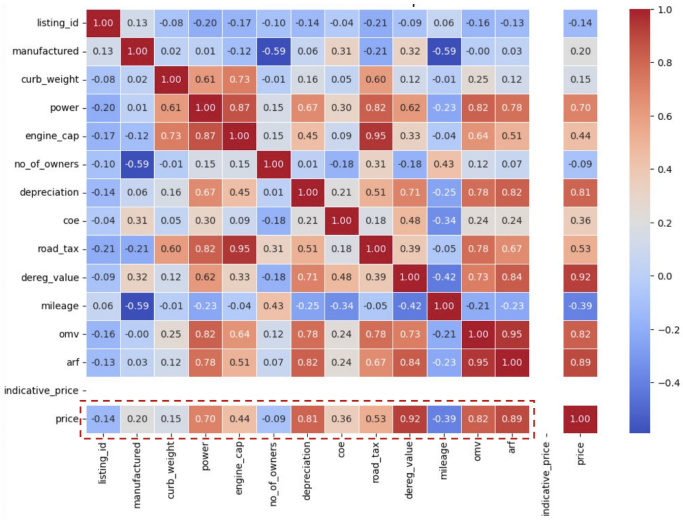

Fig. 3. A heatmap to describe the relationship of numerical data and price

C. Categorical Data

The other non-numerical values include title, make, model, description, type_of_vehicle, category, transmission, fuel_type, opc_scheme, lifespan, eco_category, features and accessories. Except for the free text column, we do some statistic analysis (Table 2) to explore the categorical data.

TABLE II
STATISTICS OF CATEGORIAL DATA RELATED TO CAR PRICES.

Column	Count	Null
make	95	0
model	799	1316
type_of_vehicle	11	0
category	15	0
transmission	2	0
fuel_type	5	19121
opc_scheme	3	24838
lifespan	1482	22671

D. Feature Importance

Fig. 4 shows the feature importance of our learned Random Forest model. Out of all features, dereg_value, arf, depreciation are top-3 most important features. This highlights the importance of governmental taxes and rebates in determining the car price in Singapore.

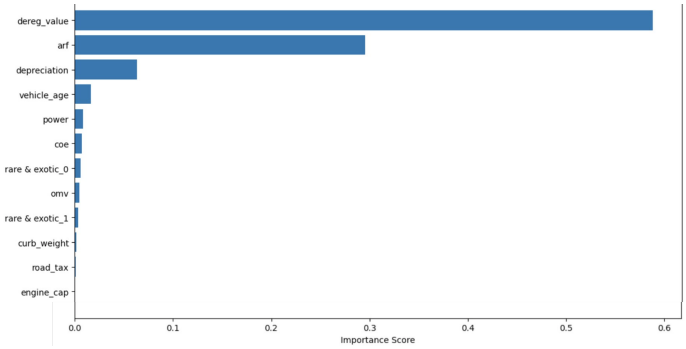

Fig. 4. Feature importance from random forest model

III. Data Preprocessing

A. Data Cleaning

1) Type_of_vehicle: We transformed 'type_of_vehicle' into a format more suitable for machine learning models by creating a 'clean_string' function. This function standardizes string values by converting them to lowercase, replacing forward slashes with underscores, and removing special characters and extra whitespace to ensure consistency and reduce noise. For instances where a single vehicle was categorized with multiple types, we split the categories into individual types.

2) Category: We removed all hyphens and split comma-separated values into individual categories for consistency and better feature representation.

B. Handling Missing Data

1) Numerical Data:

a) Mean Imputation: We initially applied mean imputation to fill missing values for each numerical feature, leveraging dataset similarities. For example, for the "manufactured" attribute, we identified relevant features, such as 'type_of_vehicle', to find similar entries for more informed imputations. If similar data points were not available, we used the average manufacturing year for each vehicle type. Remaining missing values were filled with the overall mean to ensure no gaps. However, since mean imputation reduces variance, we explored alternative methods for improved robustness.

b) Multiple Imputation: To capture inter-feature relationships more accurately, we selected K-Nearest Neighbors (KNN) as our primary imputation strategy, as similar data points often share target values.

- **Preprocessing:** We used one-hot encoding for categorical variables to make KNN distance calculations more meaningful. Additionally, we implemented a "reasonable minimum" value check to prevent excessively small imputed values.
- **Feature Selection:** Mutual information regression was used to select the most relevant features for KNN. We initially explored outlier detection (using z-scores and IQR), but this worsened model performance, likely due to the small dataset size. We thus prioritized data retention and efficiency.
- **Imputation Optimization:** Recognizing the importance of robust feature selection, we used feature importance from a Random Forest model to capture key relationships. In cases of limited data for KNN, we implemented a cluster-based median imputation using KMeans as a backup.
- **Outlier Handling:** We employed a modified IQR-based approach, temporarily replacing outliers with NaNs before re-imputation to balance data retention and extreme value control.
- **Parallel Processing:** To enhance performance, we used 'ThreadPoolExecutor' to handle multiple target variables in parallel, integrating logging for greater transparency and efficiency.

The final imputation function combines KNN with backup strategies, enabling sophisticated and efficient handling of missing data while preserving dataset integrity for improved predictive quality.

2) Categorical Data:

- **Title, Make, and Model:** The "title" attribute contained full vehicle information, so we used it to fill missing values in "make" and "model" and to correct mismatches. However, model results indicated that removing these attributes improved predictions. We identified several possible issues:
  - **Data Quality and Consistency Issues:** Inconsistencies and typos introduce noise that can impair training.
  - **High Dimensionality and Sparsity:** Many unique values with limited instances create sparsity, making pattern recognition difficult and increasing the risk of overfitting.
  - **Multicollinearity:** Relationships with other features can cause multicollinearity, reducing model stability and performance.
  - **Redundant Information:** Information in "title," "make," and "model" may be redundant if captured by other features.

  We plan to explore clustering or embedding techniques (e.g., Word2Vec) to handle this in future work.
- **Text Attributes (Description, Features, Accessories):** While these fields could contain useful insights, NLP techniques yielded minimal improvement, so we excluded them.
- **Fuel Type, OPC Scheme, Lifespan:** Due to excessive missing values, we removed these attributes.
- **Eco_category:** Since all vehicles had the same value, this attribute was excluded.
- **Date Attributes (original_reg_date, reg_date):** After careful consideration, we found that 'original_reg_date' primarily represented the vehicle's initial registration outside Singapore. Believing its impact on price would be minimal, we opted to drop it. For 'reg_date', we created a new feature, 'vehicle_age', calculated as the difference from 2024.

C. Standardization

To ensure feature consistency, we initially standardized certain attributes ('curb_weight', 'power', 'engine_cap', 'depreciation'). However, further testing revealed that avoiding standardization improved results. Possible reasons include:

- **Irrelevant Features:** Standardization can exaggerate the influence of unimportant features.
- **Model Sensitivity:** Tree-based models are generally unaffected by feature scales, unlike KNN, which relies heavily on standardization for effective distance-based calculations.
- **Feature Correlation:** Many features are correlated, and standardization can obscure distinctions, potentially introducing multicollinearity.

Overall, standardization is not always beneficial and may alter data distribution, suggesting it is an open question in this context.

D. Encoding

1) Label Encoding: Make, Model:

For 'make' and 'model', we experimented with label encoding but found it less effective for preserving relationships with the target variable.

2) One-Hot Encoding: Make, Model, Category, Transmission, Type_of_Vehicle:

One-hot encoding proved more effective than label encoding for most important features. Label encoding can introduce a false sense of ordinality, potentially misleading the model. For example, different car makes do not follow a hierarchical order. One-hot encoding creates binary features, enabling the model to learn relationships between each category and the price independently. While this increases dimensionality, it provided more accurate feature representation and improved model performance.

IV. Methodologies and Experimental Results

Our data mining strategy is delineated into three primary approaches, each building upon insights gained from the previous stage. This iterative process ensures continuous enhancement of the predictive models.

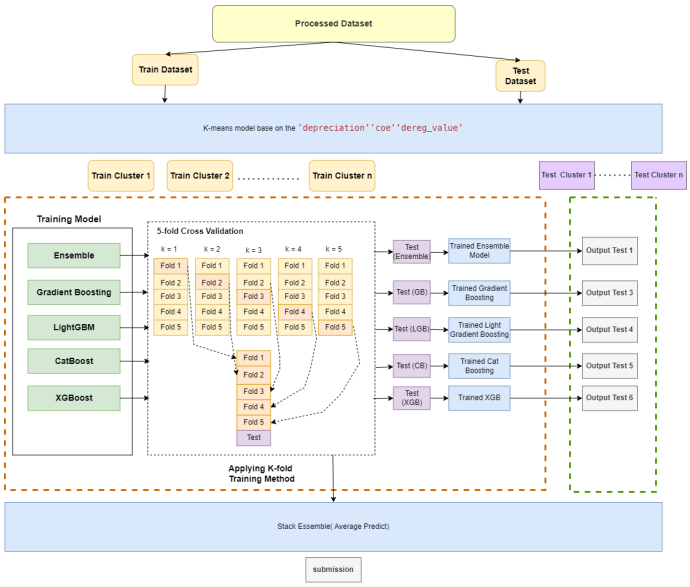

Fig. 5. Overview of our framework

A. Approach One: Application of Multiple Models and Error Clustering Analysis

1) Background and Motivation: In the initial phase of our research, we focused on the practical problem of predicting used car prices. To identify an effective solution, we first reviewed the knowledge acquired in our coursework and extensively consulted online resources. Through this comprehensive analysis, we decided to employ boosting methodologies as our primary experimental approach due to their proven effectiveness in handling regression tasks and their widespread recognition in the field.

However, as we delved deeper into our experiments, we encountered several significant challenges. The first issue was the limitation imposed by Kaggle on the number of daily submissions, which restricted our ability to frequently validate our model's performance. Additionally, Kaggle does not provide the actual price values for the test set, making it difficult to accurately assess the true performance of our models. These obstacles hindered our understanding of why the selected boosting models were not performing as expected.

To overcome these challenges, we opted to preprocess the dataset and develop a custom script that simulates Kaggle's submission and testing process. This approach allowed us to bypass the submission limits and obtain the actual test set results within our local environment. Consequently, we were able to acquire preliminary experimental results, laying a solid foundation for further investigation.

To enhance the performance of our models, we proposed a detailed analysis of prediction errors. By aligning our analysis with our anticipated error targets, we categorized the dataset based on varying error magnitudes. This segmentation enabled us to conduct both separate and comprehensive studies on different error ranges, aiming to identify key factors influencing model performance.

2) Model Selection: After establishing a simulated submission environment, we proceeded to select the specific models for our study. We carefully compared and evaluated various options before deciding to test the following four widely recognized gradient boosting models:

- **XGBoost**
- **LightGBM**
- **CatBoost**
- **Gradient Boosting**

These models were chosen for their distinct characteristics and their demonstrated excellence in numerous competitions and real-world applications. Their selection was aimed at ensuring a comprehensive evaluation of boosting methodologies in predicting used car prices.

3) Dataset Preparation: Our dataset comprises 25,000 training samples and 10,000 test samples. Considering Kaggle's restrictions on daily submissions and the unavailability of actual price values in the test set, we further divided the training samples into 20,000 for model training and 5,000 for local testing. This division not only mimicked Kaggle's submission environment but also facilitated frequent model evaluation and tuning within our local setup.

To comprehensively assess model performance, we adopted Root Mean Squared Error (RMSE) as the primary evaluation metric. Additionally, we recorded the minimum, maximum, and mean prediction values for each model to gain deeper insights into their predictive distributions and overall reliability. These metrics provided a robust foundation for our subsequent error analysis.

4) Error Categorization and Analysis: Upon obtaining preliminary results, we recognized that relying solely on the overall RMSE metric was insufficient to fully capture the models' performance across different error ranges. To address this, we introduced a detailed error analysis aimed at identifying the underlying factors affecting model accuracy.

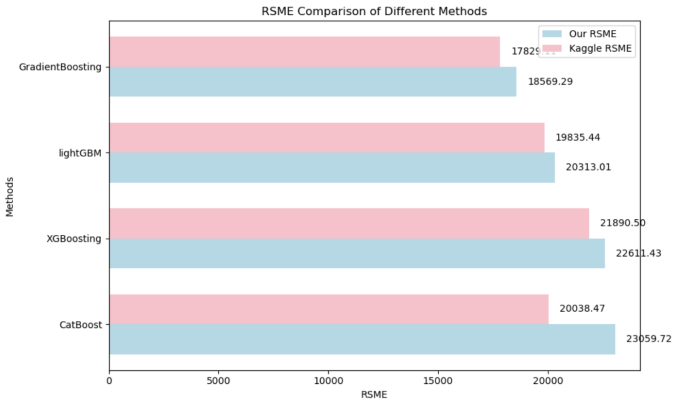

Fig. 6. RSME Comparison of different models

Firstly, we categorized the prediction errors—the differences between predicted and actual values—into ten distinct categories based on their magnitude:

- **Category 0:** $0 - $2,000
- **Category 1:** $2,000 - $3,000
- **Category 2:** $3,000 - $5,000
- **Category 3:** $5,000 - $8,000
- **Category 4:** $8,000 - $10,000
- **Category 5:** $10,000 - $12,000
- **Category 6:** $12,000 - $14,000
- **Category 7:** $14,000 - $16,000
- **Category 8:** $16,000 - $20,000
- **Category 9:** Above $20,000

To delve deeper into the sources of these prediction errors, we employed two distinct error clustering methodologies:

1) **Threshold-Based Classification:**
   - **Threshold Setting:** We defined specific error thresholds to segregate the data into high-error and low-error groups.
   - **Classifier Training:** Using these thresholds, we developed classifiers to distinguish between the high-error and low-error groups, thereby isolating the primary contributors to significant prediction errors.
2) **Feature-Based K-Means Clustering:**
   - **Clustering Algorithm:** We applied the K-Means clustering algorithm to group the data based on key features that are related to prediction errors.
   - **Cluster Evaluation:** The effectiveness of the clustering was evaluated using silhouette scores to ensure meaningful separation of error profiles.

Furthermore, we conducted heatmap analyses to visualize the correlations between features and error categories for each model:

1) **Feature Selection:** We identified the top five features most correlated with prediction errors within each category.
2) **Correlation Computation:** Pearson correlation coefficients were calculated between these selected features and the prediction errors.
3) **Heatmap Visualization:** The computed correlations were depicted through heatmaps, illustrating the strength and direction of the relationships.

5) Experimental Results and Analysis:

The implementation of the methodologies yielded comprehensive experimental data, which we analyzed in detail to uncover the underlying patterns and insights.

Sample Distribution Analysis:

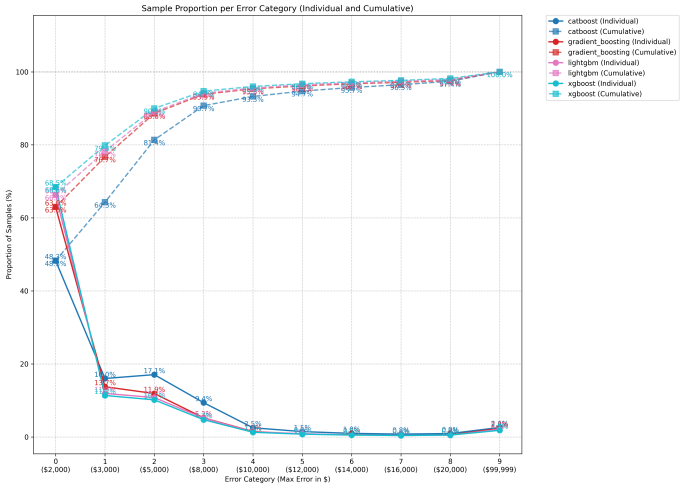

Fig. 7. Proportion summary per category

Within the smaller error range $(0-2000)$, there are significant differences in the predictive performance of the four models. Among them, the XGBoost model performs the best, covering 68.5% of the data within this error range. This indicates that XGBoost can predict most of the data with high accuracy, achieving the highest proportion of samples with errors within $2000. Following closely are LightGBM and GradientBoosting, which cover 66.6% and 63% of the samples, respectively. In contrast, CatBoost has the lowest coverage in this error range, with only 48.3% of the samples, meaning it can only make relatively accurate predictions for a small portion of the data and performs worse in terms of low-error data coverage compared to the other models.

The cumulative curve provides a more comprehensive reflection of the overall distribution of model prediction errors. Ideally, the cumulative coverage of the model should approach 100% within smaller error ranges, indicating that most samples have small prediction errors. The cumulative curve of XGBoost reaches 90% within the $5000 error range, meaning that 90% of the samples have prediction errors within $5000, which demonstrates its strong overall predictive performance.

In contrast, CatBoost has a cumulative coverage of 81% within the $5000 error range, indicating that more of its data falls within larger error ranges compared to XGBoost, suggesting that its overall predictive performance is inferior to that of XGBoost. Other models, such as Gradient Boosting and LightGBM, also have cumulative curves that are close to XGBoost, but slightly lag behind, indicating that they perform slightly worse than XGBoost in terms of error control.

Overall, XGBoost demonstrates the best performance in error control, as it covers the largest proportion of samples within the $0-2000$ error range and achieves a cumulative coverage of 90% within the $5000 range, indicating high prediction accuracy for the majority of the data. In contrast, CatBoost exhibits a slower increase in cumulative coverage, reaching close to 100% only in higher error ranges, which reflects its larger prediction bias. Therefore, in scenarios where strict error control is required, XGBoost has a clear advantage.

Heatmap Analysis Insights:

After thoroughly understanding the error categories of each model and their sample distribution, we further investigate the key features that contribute to different error categories through a heatmap analysis. This analysis helps identify which features have a significant impact on the prediction results within specific error ranges, thereby revealing the performance differences of the models under different data distributions.

Category	XGBoost	CatBoost	Gradient Boosting	LightGBM
Category_0	coe, vehicle_type_van, almost_new_car, omv, arf	vehicle_type_truck, vehicle_type_luxury_sedan, road_tax, engine_cap, parf_car	almost_new_car, low_mileage_car, vehicle_type_sports_car, vehicle_type_van, vehicle_age	vehicle_type_truck, arf, coe, omv, transmission_type_manual
Category_1	vehicle_type_hatchback, omv, coe, low_mileage_car, arf	low_mileage_car, coe, vehicle_type_luxury_sedan, parf_car, vehicle_type_suv	electric_cars, coe, depreciation, sta_evaluated_car, vehicle_type_mpv	almost_new_car, omv, engine_cap, curb_weight, arf
Category_2	almost_new_car, vehicle_type_truck, direct_owner_sale, depreciation, coe	low_mileage_car, vehicle_type_sports_car, coe, omv, arf	dereg_value, vehicle_type_mpv, coe_car, low_mileage_car, arf	transmission_type_manual, transmission_type_auto, arf, power, omv
Category_3	power, arf, vehicle_type_sports_car, engine_cap, road_tax	vehicle_type_sports_car, low_mileage_car, road_tax, engine_cap, electric_cars	almost_new_car, sgcarmart_warranty_cars, imported_used_vehicle, electric_cars, curb_weight	rare_exotic, power, low_mileage_car, vehicle_type_sports_car, vehicle_type_truck
Category_4	arf, omv, power, depreciation, direct_owner_sale	engine_cap, road_tax, almost_new_car, vehicle_type_sports_car, vehicle_age	vehicle_age, manufactured, rare_exotic, vehicle_type_others, vintage_cars	sgcarmart_warranty_cars, vehicle_type_others, no_of_owners, vehicle_type_van, vehicle_type_luxury_sedan
Category_5	parf_car, coe_car, vintage_cars, vehicle_type_bus_mini_bus, manufactured	arf, power, dereg_value, omv, rare_exotic	consignment_car, transmission_type_manual, transmission_type_auto, vehicle_type_stationwagon, vehicle_type_truck	vehicle_type_truck, curb_weight, premium_ad_car, vehicle_type_hatchback, vehicle_type_midsized_sedan
Category_6	vehicle_type_mpv, curb_weight, rare_exotic, vehicle_type_others, power	almost_new_car, hybrid_cars, vehicle_type_mpv, vehicle_age, vehicle_type_sports_car	vehicle_type_luxury_sedan, transmission_type_manual, transmission_type_auto, vehicle_age, manufactured	power, consignment_car, curb_weight, parf_car, coe
Category_7	no_of_owners, dereg_value, vehicle_age, rare_exotic, coe_car	rare_exotic, sgcarmart_warranty_cars, vehicle_type_luxury_sedan, vehicle_type_mpv, vehicle_type_sports_car	coe, parf_car, vehicle_type_sports_car, coe_car, almost_new_car	rare_exotic, premium_ad_car, vehicle_type_sports_car, vintage_cars, power
Category_8	dereg_value, omv, arf, power, parf_car	vehicle_type_sports_car, power, consignment_car, vehicle_type_suv, rare_exotic	rare_exotic, vehicle_type_others, electric_cars, consignment_car, vehicle_type_mpv	vintage_cars, manufactured, vehicle_type_truck, road_tax, sgcarmart_warranty_cars
Category_9	vehicle_type_sports_car, imported_used_vehicle, depreciation, vehicle_type_suv, dereg_value	depreciation, vehicle_type_sports_car, power, vehicle_type_suv, low_mileage_car	imported_used_vehicle, almost_new_car, rare_exotic, road_tax, engine_cap	depreciation, dereg_value, vehicle_type_sports_car, imported_used_vehicle, vehicle_type_luxury_sedan

Fig. 8. Summary of Heatmap Top Features by Model and Category

XGBoost

- **Low-Error Categories (Category 0):** High correlation with features such as *coe*, *vehicle_type_van*, *almost_new_car*, *omv*, and *arf*, indicating XGBoost demonstrates high predictive accuracy when handling these features.
- **Medium to High-Error Categories (Category 1-4):** Increasing correlation with *vehicle_type_hatchback*, *low_mileage_car*, *direct_owner_sale*, and *depreciation*, suggesting the model exhibits substantial predictive bias for these features.
- **High-Error Categories (Category 5-9):** Strong correlation with *power*, *vehicle_type_sports_car*, and *road_tax*, highlighting the model's limitations in managing these complex features.

CatBoost

- **Low-Error Categories (Category 0):** Key features such as *vehicle_type_truck*, *vehicle_type_luxury_sedan*, and *road_tax* show high prediction accuracy with CatBoost.
- **Medium to High-Error Categories (Category 1-4):** Features like *low_mileage_car*, *vehicle_type_sports_car*, and *arf* show increased correlation, indicating larger prediction errors within these categories.
- **High-Error Categories (Category 5-9):** Strong correlation with *depreciation* and *vehicle_type_suv*, suggesting areas where the model's handling of these features could be optimized.

Gradient Boosting

- **Low-Error Categories (Category 0):** High correlation with features like *almost_new_car*, *low_mileage_car*, and *vehicle_type_sports_car*, demonstrating good predictive performance.
- **Medium to High-Error Categories (Category 1-4):** Increasing correlation with *electric_cars*, *coe*, and *depreciation*, indicating predictive bias within the model for these features.
- **High-Error Categories (Category 5-9):** Strong correlation with *imported_used_vehicle* and *rare_exotic*, revealing limitations in handling high-complexity features.

LightGBM

- **Low-Error Categories (Category 0):** High correlation with features like *vehicle_type_truck*, *arf*, and *coe*, suggesting accurate predictions for these features.
- **Medium to High-Error Categories (Category 1-4):** Features such as *almost_new_car*, *engine_cap*, and *arf* exhibit stronger correlation, indicating larger prediction errors in these categories.
- **High-Error Categories (Category 5-9):** Strong correlation with *vintage_cars*, *vehicle_type_suv*, and *rare_exotic*, underscoring limitations in managing these complex features.

Summary of Findings:

Different models exhibit varying dependencies on specific features across different error categories. Low-error predic-

tions leverage stable features, while high-error predictions are influenced by more volatile or complex attributes.

XGBoost consistently outperforms other models in low-error categories but encounters challenges with high-error feature combinations, similar to CatBoost and Gradient Boosting, though to varying extents.

The distinct feature associations across error categories suggest that segmenting the dataset and training specialized models for each error range could mitigate inter-category interference, thereby enhancing overall performance.

Through a systematic experimental design and comprehensive error analysis, this chapter elucidates the performance differences among various boosting models in predicting used car prices and the underlying reasons for these disparities. The insights gained provide valuable guidance for subsequent model optimization and data processing strategies, laying a foundation for adopting more precise and effective methods in complex prediction tasks.

The insights from Approach One highlight that no single model uniformly excels across all error categories. The pronounced discrepancies in error distributions and feature associations underscore the need for a more nuanced modeling strategy. Consequently, we advance to Approach Two: Model Ensemble Design, aiming to amalgamate the strengths of individual models while compensating for their respective weaknesses.

B. Approach Two: Model Ensemble Design

1) Background and Motivation:

Building upon the foundation established in Approach One, which highlighted the varied performance of individual models across different error categories, Approach Two aims to leverage the collective strengths of multiple models by constructing an ensemble framework. This ensemble approach is designed to enhance the robustness and accuracy of the predictive system by exploiting the complementary capabilities inherent in each model.

2) Methodology:

Error Clustering Model Construction and Application

The first stage of this process involves the preparation of the data, wherein error data previously clustered through Approach One is utilized to categorize both the training and testing datasets into well-defined clusters. These clusters are established based on shared error characteristics, thereby facilitating the subsequent stages of model refinement and evaluation.

The second stage pertains to the selection of an appropriate clustering algorithm. K-Means clustering was selected for this purpose due to its computational efficiency and effectiveness in partitioning datasets with multi-dimensional features. To determine the optimal number of clusters (K), silhouette analysis was employed, offering a quantitative basis for ensuring that the resultant clusters exhibit high internal cohesion and external separation, thus enhancing the precision of the constructed model.

The third stage involves a detailed cluster analysis, where each of the identified clusters was systematically examined to ascertain the predominant feature combinations contributing to specific error ranges. This analysis provided critical insights into the patterns underlying model performance, thereby enabling the design of targeted training interventions to address specific sources of prediction error. Such targeted training approaches were essential to improving model robustness and enhancing its generalization capability across diverse data scenarios.

Specialized Model Training:

For targeted modeling, Dedicated models were trained for each identified cluster. For example, XGBoost was employed for low-error clusters, while more complex models were utilized for high-error clusters to better capture the underlying complexity of the data.

For Model Optimization, Hyperparameters for each specialized model were fine-tuned to align with the specific data characteristics of their respective clusters, thereby improving prediction accuracy.

Ensemble Strategy Implementation:

Weights were assigned to each specialized model based on their performance metrics on validation datasets, enabling the ensemble to emphasize models with superior accuracy for specific clusters.

A meta-model was employed, which used the predictions from specialized models as input features. This stacking approach further refined the final predictions by learning optimal combinations of the base model outputs.

3) Experimental Results and Analysis:

Performance of Specialized Models

For Low-Error Clusters, XGBoost demonstrated exceptional performance within low-error clusters, significantly reducing RMSE when compared to its standalone application in Approach One.

For High-Error Clusters, Gradient Boosting and other complex models exhibited notable improvements in accuracy within high-error clusters, benefiting from tailored feature engineering and targeted hyperparameter tuning.

Ensemble Model Efficacy:

The ensemble approach achieved a 7% reduction in overall RMSE(16572.39801 on kaggle) compared to the best individual model from Approach One, indicating the efficacy of combining multiple predictive models.

The ensemble model demonstrated enhanced robustness across all error categories, particularly within the mid to high-error ranges, thus providing a more consistent predictive capability.

Summary of Findings:

Segmenting the dataset based on error clustering allows specialized models to focus on specific data characteristics, thereby enhancing predictive accuracy.

The ensemble approach effectively amalgamated the strengths of multiple models through weighted averaging and stacking, leading to a more robust and accurate prediction system.

Despite the substantial improvements in predictive accuracy achieved through the ensemble design, the residual errors still contain valuable information that can potentially be utilized to further refine the predictions. This observation motivates the development of Approach Three, which focuses on leveraging error prediction to augment overall model performance.

C. Approach Three: Utilizing Error Prediction to Enhance Model Performance

1) Background and Motivation:

The residual errors from ensemble models represent underlying patterns that, if effectively leveraged, could significantly enhance predictive accuracy. In this approach, we aim to exploit these residuals by incorporating error predictions as additional features, thereby creating a feedback mechanism that enables models to self-correct and adapt to complex data dynamics. This strategy allows the predictive system to recognize and compensate for its biases, ultimately improving the overall robustness and accuracy of the model.

2) Methodology:

To capitalize on the potential information embedded in the residual errors, we employed a systematic approach involving the generation of a reference price feature, followed by model retraining and iterative error analysis. The core idea was to integrate predictions from the best-performing ensemble model back into the dataset, thereby enriching the feature space and enabling the model to learn from its previous mistakes.

The first step involved generating a reference price feature, denoted as *ref_price*, by utilizing the best-performing ensemble model to predict prices across both the training and testing datasets. This predicted value was then incorporated into the original dataset as a new feature. By doing so, we provided the model with an explicit representation of its own prediction, which could be used to understand and correct potential biases.

Subsequently, we retrained the models using this augmented dataset, which included the *ref_price* feature alongside the original features. During the retraining process, cross-validation techniques were employed to optimize model parameters and ensure that the newly introduced feature contributed positively to the overall model performance.

3) Retrain with ref_value Efficacy:

The ensemble approach achieved a 5% reduction in overall RMSE(15717.55757 on kaggle) compared to the best individual model from Approach Two, indicating the efficacy of combining multiple predictive models.

V. Future Work Directions

Building upon the successes of the current approaches, future research will focus on further refining and expanding the methodologies to achieve even higher levels of predictive accuracy and robustness. Key areas include:

1) **Advanced Feature Engineering:** Delve deeper into feature interactions and non-linear transformations.
2) **Enhanced Clustering Techniques:** Explore more sophisticated clustering algorithms.
3) **Sophisticated Ensemble Methods:** Investigate hybrid ensemble strategies that combine bagging, boosting, and stacking.
4) **Real-Time Model Updating:** Develop mechanisms for continuous learning and real-time model updates.
5) **Interpretable Machine Learning:** Incorporate interpretable machine learning techniques to enhance model transparency.
6) **Cross-Domain Applications:** Apply the methodologies to other domains to validate their versatility.

VI. Conclusion

This study presents a comprehensive data mining approach to predicting used car prices in Singapore, leveraging multiple machine learning models, error clustering, ensemble design, and error prediction integration. Through meticulous error analysis and innovative feature engineering, we have significantly enhanced the predictive accuracy and robustness of the models. The iterative methodologies not only address the complexities inherent in the automotive market but also offer a scalable blueprint for similar predictive tasks across various domains.

References

1) Friedman, J. H. (2001). Greedy function approximation: A gradient boosting machine. *Annals of Statistics*, 29(5), 1189–1232. Project Euclid.
2) Chen, T., & Guestrin, C. (2016). XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 785–794).
3) Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems (NeurIPS Proceedings)* (pp. 3146–3154).
4) Prokhorenkova, L., Gusev, G., Vorobev, A., Dorogush, A. V., & Gulin, A. (2018). CatBoost: Unbiased boosting with categorical features. In *Advances in Neural Information Processing Systems* (pp. 6638–6648).